Talking Drums: Generating drum grooves with neural networks

P. Hutchings[*1]

[1]Monash University, Melbourne, Australia

Presented is a method of generating a full drum kit part for a provided kick-drum sequence. A sequence to sequence neural network model used in natural language translation was adopted to encode multiple musical styles and an online survey was developed to test different techniques for sampling the output of the softmax function. The strongest results were found using a sampling technique that drew from the three most probable outputs at each subdivision of the drum pattern but the consistency of output was found to be heavily dependent on style.

Keywords: RNN, percussion, generative music, translation

1 Introduction

This research details the development of a percussion-role agent as part of a larger project where virtual, self-rating agents with different musical roles work in a process of co-agency to generate music compositions in real-time [Hutchings and McCormack, 2017]. The percussion-role agent was developed for generating multiple possible multi-instrument percussion parts to accompany provided melodies and harmonies in real-time.

A neural network based agent was developed to incorporate a range of different music styles from a large corpus of compositions and to utilise a softmax function as part of the self-rating process. A network architecture used in natural language translation was adopted based on the idea that a percussion score could be considered as containing multiple drums 'speaking' different languages but saying the same thing at the same time. The network was trained on a collection of drum kit scores from over 250 pop, rock, funk and Afro-Cuban style compositions and patterns from drum technique books. The output of the network was evaluated from an online survey and a physical interface was developed for feeding kick-drum parts into the network.

---

[*]pehut2@student.monash.edu

1.1 Related work

Markov models [Hawryshkewich et al., 2010] [Tidemann and Demiris, 2008], generative grammars [Bell and Kippen, 1992] and neural network models Choi et al. [2016] have all been shown to be effective in the area of drum score generation. The approach shown in this paper is based on the requirements of generating an agent for a multi-agent composition system. Research in this area has demonstrated the need for agent models to match the needs of the overall system [Eigenfeldt and Pasquier, 2009].

The similarities and differences between music and natural language have been explored in detail [Patel, 2003] [Mithen, 2011]. While distinct differences exist in terms of cognitive processing, semantics and cultural function, there are similarities in the structure of phrases that have lead to the use of natural language processing techniques in the analysis and generation of music.

1.2 Translation model

Generating a full drum kit score based on the rhythm of one or more individual instruments in the kit is a problem with different challenges than natural language translation. All translations are one to one in word count. Music is a non-semantic form of communication which allows for and values greater structural variation than spoken language so imperfect translations can still be effective. Conversely because there is no perfect translation, there are many different outputs for a given input in the training data, decreasing convergence during training. The problem can also be viewed as one of data-expansion as a single instrument part is expanded to fill a full drum kit with multiple concurrent instruments being used. To take advantage of these strengths and diminish the weaknesses of a translation based neural network model a new syntax for expressing drum parts was developed.

2 Method

2.1 Data preprocessing

A collection of 250 drum kit scores in 4/4 were found on drum tablature websites and books and parsed into a music-XML format. Tracks were selected based on the most viewed webpages for rock, pop, funk and Afro-Cuban styles of music and were each checked for accuracy by comparing with the original recordings by ear. Pop, rock and funk styles were selected due to their global popularity and typical use of a standard drum kit. The Afro-Cuban style was added to this list to see if some of the stricter idiomatic structures of the style, such as the 'clave' rhythmic pattern, could be preserved. Afro-Cuban and funk drum tablatures were more difficult to find so the tablatures were augmented with patterns from drum technique instruction books. For each genre a total of 7000-7500 bars were parsed.

Each bar was divided into 48 subdivisions, allowing all triplet and tuple divisions down to the resolution of semiquaver triplets to be represented. Each division was given a word token that represented the drums being hit on that subdivision and barlines were replaced with a word token describing the musical style which allowed multiple styles to be encoded in a single RNN network.

The tokenised phrase in Equation 1 represents a kick-drum being kicked on each beat of a

single 4/4 bar and a 'pop' style description.

$$pop \; K \; o \; o \; o \; o \; o \; o \; o \; o \; o \; o \; o \; K \; o \; o \; o \; o \; o \; o \; o \; o \; o \; o \; o \; K \; o \; o \; o \; o \; o \; o \; o \; o \; o \; o \; o \; K \; o \; o \; o \; o \; o \; o \; o \; o \; o \; o \; o \quad (1)$$

The full list of letter representations used to create word tokens are presented in Table 1. Composition segments of 4 bars were used as sentences for training the neural network with kick-drum patterns used as inputs to the encoder layer and the rest of the drum parts in the decoder layer. Encoder input sequences were reversed and encoded using one-hot encoding. The kick-drum was selected as the input language because it is usually used to mark the beat of a composition and small changes can dramatically affect the feeling of time.

Table 1: Letter representations of drums

Drum	Cymbal	Hi-hat	Snare	High Tom	Tom	Floor Tom	Kick	None
Letter	C	H	S	T	t	F	K	o

2.2 Network architecture

The neural network has an RNN sequence-to-sequence architecture [Sutskever et al., 2014] using the Tensorflow deep-learning framework [Abadi et al., 2015]. A model layer of size 128 and 3 layers produced a perplexity of 1.15 when trained with a learning rate of 0.55 and a gradient descent optimiser. This was the lowest perplexity achieved from a manual testing of variations to these hyper-parameters. Hidden states were initialised with all zero values and updated at each step of training.

3 Evaluation

An online survey was generated to find a sampling technique that human listeners found preferable. The survey was advertised on social media groups related to drumming and computer music and run for two weeks.

3.1 Survey

Participants were presented with a style menu and a 48 step sequence with an editable kick-drum line that they could use to design a four beat kick-drum pattern as seen in Fig. 1. After clicking a 'Generate Groove' button on the interface, the other instrument parts would be generated and a loop of the pattern would begin playing with sounds sampled from drum kits. Participants were then asked to rate the groove as poor, average or good. The survey was designed to encourage a fast and playful experience, so demographic data was not asked or collected.

Each time a groove was generated the web application ran the input through the neural network and randomly selected a sampling method. Three sampling methods were tested: A greedy decoder (Method 1), a roulette-wheel sampler across all probabilities (Method 2) and a roulette-wheel sampler of the three most probably tokens at each subdivision (Method 3).
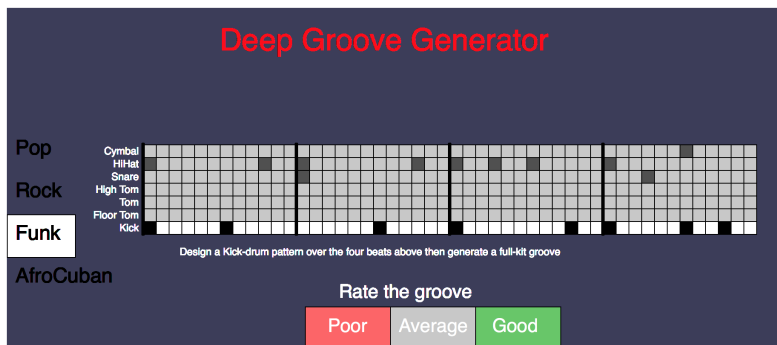
Figure 1: Interface for the online evaluation survey

3.2 Results

A total of 1278 groove evaluations were recorded in the survey.

Table 2: Survey results for different sampling methods

	Raw			Normalised		
	Good	Average	Poor	Good	Average	Poor
Method 1	91	276	30	0.23	0.70	0.08
Method 2	100	217	125	0.23	0.49	0.28
Method 3	172	183	84	0.39	0.42	0.19

As shown in Table 2 the model produced full drum-kick patterns that were deemed to be average or good in a majority of ratings on the web survey. Of the three sampling methods it can be observed that the greedy encoder had a tendency towards results that participants deemed average. The roulette wheel sampling used in Method 2 had the highest rate of 'poor' ratings. Overall the best performer was the sampler that drew from the three most probable tokens at each subdivision. Examples of 5 drum patterns for each sampling method are available to listen to at https://doi.org/10.6084/m9.figshare.4903181.v1.

Table 3: Mean rating for mean initial probabilities of selected notes.
Poor =0, average =1, good = 2

Mean probability	0.2-0.3	0.3-0.4	0.4-0.5	0.5-0.6	0.6-0.7	0.7-0.8	0.8-0.9
Mean rating	0.25	0.27	0.58	1.14	1.32	1.54	1.22

4 Discussion and future work

The ratings in Table 3 peaked when the average probability was between 0.7-0.8, below the maximum observed bracket of 0.8-0.9. This may be a result of participants valuing familiar but different drum patterns over patterns that they may have heard in songs they know. The significantly higher rating of one band of probability range supports the use of the model in the intended application of a multi-agent system as it provides a means of self-rating output. Mean

ratings of Afro-Cuban style patterns were significantly lower (24% poor) than for other styles (16-18% poor) which may be the result of stylistic bias of the participants or could suggest important elements of the style are not represented in the model output.

A syntax for expressing desired accents is being developed as an encoder to expand the pallet and may improve results in the Afro-Cuban and other styles. A physical drum-pedal interface has been developed to test the system with drummers in a natural playing position.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org.

Bernard Bell and Jim Kippen. Bol processor grammars, understanding music with ai: perspectives on music cognition, 1992.

Keunwoo Choi, George Fazekas, and Mark B. Sandler. Text-based LSTM networks for automatic music composition. *CoRR*, abs/1604.05358, 2016. URL http://arxiv.org/abs/1604.05358.

Arne Eigenfeldt and Philippe Pasquier. A realtime generative music system using autonomous melody, harmony, and rhythm agents. In *XIII Internationale Conference on Generative Arts, Milan, Italy*, 2009.

Andrew Hawryshkewich, Philippe Pasquier, and Arne Eigenfeldt. Beatback: A real-time interactive percussion system for rhythmic practise and exploration. In *NIME*, pages 100–105, 2010.

Patrick Hutchings and Jon McCormack. Using autonomous agents to improvise music compositions in real-time. In *International Conference on Evolutionary and Biologically Inspired Music and Art*, pages 114–127. Springer, 2017.

S. Mithen. *The Singing Neanderthals: The Origins of Music, Language, Mind and Body*. Orion, 2011. ISBN 9781780222585. URL https://books.google.com.au/books?id=3aP0tkerd_wC.

Aniruddh D Patel. Language, music, syntax and the brain. *Nature neuroscience*, 6(7):674–681, 2003.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Axel Tidemann and Yiannis Demiris. A drum machine that learns to groove. In *Annual Conference on Artificial Intelligence*, pages 144–151. Springer, 2008.